**Задание 2.**

**Изучите данные и выполните следующие задачи. Выполните задание с помощью SQL.**

Клиенты оставляют заявки на звонок, после чего менеджеры звонят и обсуждают все необходимые вопросы.

Есть 2 таблицы с данными. В таблице tasks лежат заявки от клиентов, в таблице calls лежат звонки от менеджеров клиентам.

Структура таблицы **tasks:**

|  |  |  |
| --- | --- | --- |
| client\_id | created\_datetime | title |

client\_id – уникальный идентификатор клиента

created\_datetime – дата и время создания заявки клиентом

title – тема заявки

Структура таблицы **calls:**

|  |  |  |
| --- | --- | --- |
| manager\_id | client\_id | call\_datetime |

manager\_id – идентификатор менеджера, который выполнил заявку

client\_id – уникальный идентификатор клиента, связь с таблицей tasks по client\_id

call\_datetime – дата и время выполнения заявки менеджером

Напишите запросы, которые выводят:

1. Сколько заявок приходило каждый день в июне 2022 года.

**Решение:**

**SELECT date(created\_datetime) as created\_datetime**

**, COUNT(\*) as all\_requests\_day**

**FROM tasks**

**WHERE created\_datetime BETWEEN '2022-06-01' AND '2022-06-30'**

**GROUP BY created\_datetime**

**ORDER BY 1 ASC**

*Использую функцию date, для работы только с датой, без времени, чтобы при подсчете данных за день, происходила группировка по дате, без учета времени.*

*Июнь 2022 года, задаю интервалом при помощи команды BETWEEN*

*Группирую по датам при помощи GROUP BY и подсчитываем кол-во заявок на каждый день при помощи COUNT(\*)*

*Сортировка ORDER BY выполнена для удобства, чтобы даты шли по порядку.*

1. Список тем, для которых обращений было больше 10 в апреле 2022 года

**Решение:**

**SELECT title**

**, COUNT(\*) as number\_topics**

**FROM tasks**

**WHERE date(created\_datetime) BETWEEN '2022-04-01' AND '2022-04-30'**

**GROUP BY 1**

**HAVING COUNT(\*) > 10**

*Выбираем данные из таблицы tasks, задаем условие выборки по дате создания применив команду BETWEEN*

*Группируем по теме*

*Задаем фильтрацию при помощи HAVING, чтобы получить темы для которых обращений было больше 10.*

1. Список клиентов, которые оставляли заявку, но ни одного звонка от менеджера по ним не было.

**Решение:**

**SELECT DISTINCT t.client\_id**

**FROM tasks as t**

**LEFT JOIN calls as c**

**ON t.client\_id = c.client\_id**

**WHERE c.client\_id IS NULL**

*Выбираем данные из таблицы tasks, присваиваем ей псевдоним t, далее нам нужно сравнить таблицу tasks с таблицей calls (присвоим ей псевдоним c), для того чтобы узнать список клиентов, которым не звонили менеджеры будем использовать LEFT JOIN, это объединение поможет нам решить задачу.*

*Объединять будем по столбцу client\_id, т.к. по этому элементу существует связь двух таблиц.  
В условии, для получения нужной выборки, пропишем с.client\_id IS NULL*

1. Список клиентов, последние заявки которых не обработаны. Считаем, что если менеджер позвонил клиенту после заявки, то он ее обработал. Если заявок до звонка было несколько, то при звонке обрабатываются все заявки сразу.

**Решение:**

**with**

**last\_request\_client as**

**(SELECT client\_id**

**, max(created\_datetime) as last\_request**

**FROM tasks**

**GROUP BY 1),**

**ready\_request as**

**(SELECT c.\* from last\_request\_client as lr**

**join calls as c on lr.client\_id = c.client\_id**

**WHERE c.call\_datetime >= lr.last\_request)**

**SELECT client\_id**

**, max(created\_datetime) as max\_date**

**from tasks t**

**WHERE t.client\_id not in**

**(SELECT r.client\_id from ready\_request r)**

**GROUP BY 1**

*Использовал общие табличные выражения, чтобы вывести клиентов, последние заявки которых не обработаны.*

*В первом запросе* **last\_request\_client** *внутри with получаем всех клиентов и последнюю дату заявки у каждого.*

*Второй запрос* **ready\_request** *выводит все закрытые заявки*

*И итоговым запросом мы получаем клиентов, последние заявки которых не обработаны исключая закрытые заявки.*

1. Минимальную разницу между обращениями для каждого клиента, а затем вывести среднее по полученным значениям.

**Решение:**

**SELECT avg(average.minimum)**

**from (SELECT client\_id,min(svod.diff) as minimum**

**from (SELECT client\_id**

**, created\_datetime**

**, lead(created\_datetime) OVER (PARTITION BY client\_id ORDER BY created\_datetime asc) as date\_next**

**,((lead(created\_datetime) OVER (PARTITION BY client\_id ORDER BY created\_datetime asc))-created\_datetime) as diff**

**FROM tasks) as svod**

**WHERE diff is not NULL**

**GROUP BY 1) as average**

*Применил вложенные запросы*

*Для того чтобы вычислить разницу между обращениями для каждого клиента, в запросе с псевдонимом svod использовал оконную функцию lead, которая возвращает значение, стоящее после текущего.*

*Далее получаю разницу от значения полученного при помощи lead, текущее значение.*

*Создаю еще один запрос, с псевдонимом average, в котором получаю id клиента и его минимальную разницу между обращениями, отсекая строки, в которых не было расчета и они имеют значение NULL*

*В итоговом select при помощи функции avg получаем среднее по полученным значениям.*